PCT/GB 99/01582

09/674816

The Patent Office

ETKU

INVESTOR IN PEOPLE

The Patent Office
Concept House
Cardiff Road
Newport
South Wales
NP10 8QQ

REC'D 15 JUN 1999
PCT

GB99/1392

I, the undersigned, being an officer duly authorised in accordance with Section 74(1) and (4) of the Deregulation & Contracting Out Act 1994, to sign and issue certificates on behalf of the Comptroller-General, hereby certify that annexed hereto is a true copy of the documents as originally filed in connection with the patent application identified therein.

In accordance with the Patents (Companies Re-registration) Rules 1982, if a company named in this certificate and any accompanying documents has re-registered under the Companies Act 1980 with the same name as that with which it was registered immediately before re-registration save for the substitution as, or inclusion as, the last part of the name of the words "public limited company" or their equivalents in Welsh, references to the name of the company in this certificate and any accompanying documents shall be treated as references to the name with which it is so re-registered.

In accordance with the rules, the words "public limited company" may be replaced by p.l.c., plc, P.L.C. or PLC.

Re-registration under the Companies Act does not constitute a new legal entity but merely subjects the company to certain additional company law rules.

Signed

Dated 11 JUN 1999

PRIORITY DOCUMENT

SUBMITTED OR TRANSMITTED IN COMPLIANCE WITH RULE 17.1(a) OR (b)

Patents Form 1/77

s Act 1977
16)

The Patent Office

06 MAY 1998

08MAY98 E358664-1 D00192
P01/7700 25.00 - 9809685.2

Request for grant of a patent

(See notes on the back of this form. You can also get an explanatory leaflet from the Patent Office to help you fill in this form))

The Patent Office

Cardiff Road
Newport
Gwent NP9 1RH

1. Your reference ~~P.74723 SLS~~ P.74723 SLS

2. Patent application number
(The Patent Office will fill in this part) **9809685.2**

3. Full name, address and postcode of the or of each applicant *(underline all surnames)*

SONY UNITED KINGDOM LIMITED
The Heights
Brooklands
Weybridge
SURREY KT13 OXW

Patents ADP number *(if you know it)*

If the applicant is a corporate body, give the country/state of its incorporation: United Kingdom 6522780001

4. Title of the invention

5. Name of your agent *(if you have one)*

"Address for service" in the United Kingdom to which all correspondence should be sent *(including the postcode)*

J A KEMP & CO

14 SOUTH SQUARE
GRAY'S INN
LONDON WC1R 5LX

Patents ADP number *(if you know it)* 26001

6. If you are declaring priority from one or more earlier patent applications, give the country and the date of filing of the or of each of these earlier applications and *(if you know it)* the or each application number

Country	Priority application number *(if you know it)*	Date of filing *(day / month / year)*

7. If this application is divided or otherwise derived from an earlier UK application, give the number and the filing date of the earlier application

Number of earlier application	Date of filing *(day / month / year)*

8. Is a statement of inventorship and of right to grant of a patent required in support of this request? *(Answer "Yes" if:*

a) any applicant named in part 3 is not an inventor, or
b) there is an inventor who is not named as an applicant, or
c) any named applicant is a corporate body:
See note (d))

Patents Form 1/77

9. Enter the number of sheets for any of the following items you are filing with this form. Do not count copies of the same document

Continuation sheets of this form	
Description	30
Claim(s)	—
Abstract	—
Drawing(s)	—

10. If you are also filing any of the following, state how many against each item.

Priority documents	
Translations of priority documents	
Statement of inventorship and right to grant of a patent *(Patents Form 7/77)*	
Request for preliminary examination and search *(Patents Form 9/77)*	
Request for substantive examination *(Patents Form 10/77)*	
Any other documents *(please specify)*	

11. I/We request the grant of a patent on the basis of this application

Signature

Date 6 May 1998

J.A. KEMP & CO.

12. Name and daytime telephone number of person to contact in the United Kingdom 0171 405 3292

Warning

After an application for a patent has been filed, the Comptroller of the Patent Office will consider whether publication or communication of the invention should be prohibited or restricted under Section 22 of the Patents Act 1977. You will be informed if it is necessary to prohibit or restrict your invention in this way. Furthermore, if you live in the United Kingdom, Section 23 of the Patents Act 1977 stops you from applying for a patent abroad without first getting written permission from the Patent Office unless an application has been filed at least 6 weeks beforehand in the United Kingdom for a patent for the same invention and either no direction prohibiting publication or communication has been given, or any such direction has been revoked.

Notes

a) If you need help to fill in this form or you have any questions, please contact the Patent Office on 0645 500505.

b) Write your answers in capital letters using black ink or you may type them.

c) If there is not enough space for all the relevant details on any part of this form, please continue of a separate sheet of paper and write "see continuation sheet" in the relevant part(s). Any continuation sheet should be attached to this form.

d) If you have answered "Yes" Patents Form 7/77 will need to be filed.

NICAM AV/C CTS

Subunit Proposal

Table of Contents

1. INTRODUCTION 4
2. REFERENCES 5
3. CHANGES FROM PREVIOUS VERSION 6
4. NCAM CONCEPTS 7
5. CA SUBUNIT MODEL 8
5.1 THE BASIC CA SUBUNIT 8
5.1.1 *CA subunit destination plug* 8
5.1.2 *CA subunit source plug* 8
5.1.3 *Connections* 8
5.1.4 *Data Transmitted Over an Isochronous Channel* 10
5.2 CA SUBUNIT IDENTIFIER DESCRIPTOR, STATUS DESCRIPTOR, OBJECTS AND OBJECT LISTS .. 10
5.2.1 *CA Subunit Identifier* 11
5.2.2 *CA Subunit Status Descriptor* 12
5.3 CA SUBUNIT COMMANDS 13
5.3.1 *CA_ENABLE* 13
6. SMART CARD READER COMMANDS 19
6.1 REQUIREMENTS 19
6.2 COMMANDS 19
7. DATE & TIME RESOURCE 21
8. MODEM SUBUNIT 22
8.1 REQUIREMENTS 22
8.2 CONNECTIONS 22
8.3 MODEM SUBUNIT IDENTIFIER DESCRIPTOR, STATUS DESCRIPTOR, OBJECTS AND OBJECT LIST 22
8.3.1 *Modem Subunit Identifier* 23
8.3.2 *Modem Subunit Status Descriptor* 24
8.4 MODEM SUBUNIT COMMANDS 25
8.4.1 *connect_PSTN_modem_channel* 25
8.4.2 *connect_cable_modem_channel* 27
8.4.3 *disconnect_modem_channel* 28
8.4.4 *modem_status* 28
8.4.5 *modem_data_transfer* 28
9. OPERATIONAL GUIDELINES 29
10. OUTSTANDING POINTS 30

List of Figures

Figure 5-1 The Basic CA Subunit 8
Figure 5-2 CA Subunit Logical Connections 9
Figure 5-3 CA_ENABLE Control Command 14
Figure 5-4 CA_descriptor at Programme Level 16
Figure 5-5 CA_descriptor at Elementary Stream Level 17
Figure 5-6 CA_ENABLE response 17

Figure 5-7 Format of elementary_CA_enable_field 18
Figure 6-1 FUNCTION Control Command 19
Figure 6-2 SMARTCARD Status Command 20
Figure 6-3 SMARTCARD status command response 20
Figure 8-1 General Subunit Identifier Structure 23
Figure 8-2 Modem Subunit Dependent Information 23
Figure 8-3 Modem Subunit Identifier 23
Figure 8-4 Modem Status Descriptor 24
Figure 8-5 General Modem Status Format 24
Figure 8-6 connect_PSTN_modem_channel Control Command 25
Figure 8-7 connect_PSTN_modem_channel ACCEPTED Response 26
Figure 8-8 connect_PSTN_modem_channel REJECTED Response 27
Figure 8-9 connect_cable_modem_channel Control Command 27
Figure 8-10 connect_cable_modem_channel ACCEPTED Response 27
Figure 8-11 connect_cable_modem_channel REJECTED Result 28
Figure 8-12 disconnect_modem_channel Control Command 28
Figure 8-13 modem_status Status Command 28

List of Tables

Table 5-1 Allowed Connections to a CA Subunit 10
Table 5-2 General Subunit Identifier Descriptor 11
Table 5-3 CA Subunit Identifier 12
Table 5-4 System ID Codes 12
Table 5-5 Implementation Profiles 12
Table 5-6 CA Status Descriptor 12
Table 5-7 General CA Status Format 13
Table 5-8 CA Subunit Status Descriptor Identifier 13
Table 5-9 DVB CI Subunit Commands 13
Table 5-10 System ID Codes 15
Table 5-11 Definition of ca_pmt_list_management 15
Table 5-12 Definition of ca_pmt_cmd_id 16
Table 5-13 Definition of CA_enable 18
Table 5-14 Definition of result field 18
Table 6-1 Definition of smartcard_operation 19
Table 6-2 Definition of smartcard_operation 20
Table 6-3 Definition of smartcard_operation response 20
Table 8-1 Definition of modem_type 24
Table 8-2 Definition of implementation_profile_id 24
Table 8-3 modem Subunit Status Descriptor Identifier 25
Table 8-4 Definition of connection_result 26
Table 8-5 Definition of connection_result 27
Table 8-6 Definition of reason 27
Table 8-7 Definition of connection_result 27
Table 8-8 Definition of connection_result 28
Table 8-9 Definition of reason 28

1. Introduction

A requirements exists for a Conditional Access (CA) system that allows the manufacturer of a Digital Television Receiver (DTV) to access scrambled services from several broadcasters. This is achieved by defining a protocol that allows the CA system to reside on a module which can then be connected to the DTV allowing that DTV to access the service. A solution exists in the form of a PC Card connected to a single receiver. However there exists a new requirement for a Networked Conditional Access Module (NCAM). The main requirements for this device are:-

- flexible form factor
- flexible access, for example peer to peer communication
- flexible location

This document proposes the format of the additional AV/C subunits that are required to implement the NCAM. The AV/C model for the NCAM will provide a conditional access system that is tailored for use on an IEEE 1394–1995 based digital network.

2. References

[1] EN 50221 Common Interface Specification for Conditional Access and other Digital Video Broadcasting Decoder Applications
[2] ISP/IEC 13818-1 Generic Coding of Moving Picture and Associated Audio Systems.
[3] IEEE 1394-1995, Standard for a High Performance Serial Bus, 30 August 1996.
[4] AV/C Digital Interface Command Set, Version 2.0D, 26 March 1997.
[5] AV/C Tuner Model Working Specification, Version 1.0, 4 July 1997.
[6] pr ETS 300 468 "Specification for Service Information (SI) in Digital Video Broadcasting (DVB) Systems.
[7] ISO 8859: "Information processing – 8-bit single-byte coded graphic character sets, Latin alphabets".

3. Changes from Previous Version

Modified table 3-1 to reflect changes in AV/C Digital Interface Command Set General Specification, Version 3.0FC1, March 29,1998

4. NCAM Concepts

The purpose of the Networked Conditional Access Module (NCAM) is to provide conditional access functionality. The NCAM uses a logical collection of resources that allow the descrambling of selected services to take place. The required resources for the NCAM can exist either in one location, for example inside a DTV, or be distributed throughout the In Home Digital Network (IHDN).

The NCAM relies on both existing and additional subunits. The existing subunits that the NCAM makes use of are: -

- Tuner subunit
- Panel subunit

The additional subunits that are defined in this document are: -

- CA subunit
- Modem subunit
- Smart Card subunit

The Tuner subunit is used as the data source, the Panel subunit is used to provide information to the user and receive input from the user. The CA subunit contains the descrambling functionality and can make use of the smart card and modem subunits.

The modem subunit may be required when pay per view services are requested or user input is required to be returned to the service provider. The display device is used to provide a man machine interface (MMI) with the user, this interface can support both display and user input, such as a keyboard or remote control.

The resources that are required for an NCAM to function may be implemented privately within a single module. If a manufacturer wishes to develop an NCAM with the smart card and modem functionality integrated for the exclusive use of the NCAM this is allowed. In such a case the NCAM would only implement the CA subunit and make use of the tuner and panel subunits in other devices. It is likely for security reasons that an NCAM would be implemented with a private smart card. The smart card subunit is included for when a smart card could be used for other applications, for example a data card or "electronic cash" card.

The NCAM can also be implemented with distributed resources. In this case the CA subunit would work in conjunction with subunits embedded in other objects distributed throughout the digital network.

Depending on the service to be descrambled, all or some of the resources will be required. In a simple system that relies on a Smart Card to be inserted to authenticate the service the modem is not required, a simple form of display device is required to prompt the user to insert the card but interaction is not necessary. A more complicated system, for example a pay per view (PPV) system, requires all of the resources to allow a choice of services to be presented to the user and to allow the user make a selection. Therefore the NCAM may operate with reduced functionality if not all the required subunits are present.

The use of descriptors was considered for the transfer of data relating to the user programme selection for the purposes of descrambling and the connection destination for the PSTN modem but it was felt that this information was better delivered as part of a command. This is because the descriptor mechanism requires the controller to have a greater knowledge of the working mechanism of the target device. Also the use of descriptors would involve a two-stage command system, the first stage to download the data and the second to instruct the target to perform an action on the data. Sending the data with command seemed a more efficient mechanism.

5. CA Subunit Model

5.1 *The Basic CA Subunit*

The concept of the CA subunit is to provide the core functionality of the NCAM. The CA subunit would be the subunit to which the transport streams are directed and where the CA processing is carried out. The CA subunit can then communicate with other required subunits via asynchronous commands across the 1394 [3] network.

The CA subunit can be a stand-alone device or integrated into another device.

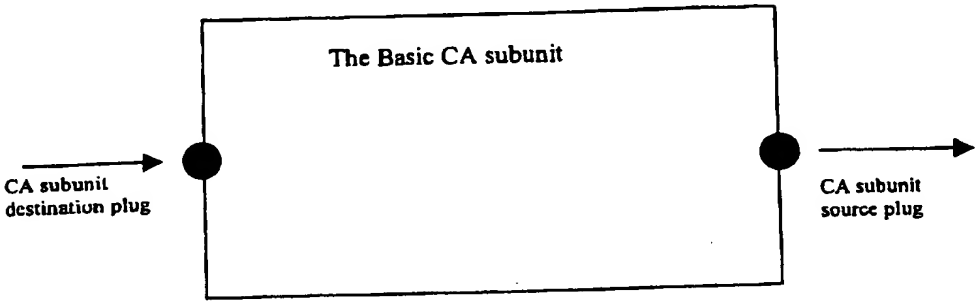

Figure 5-1 The Basic CA Subunit

5.1.1 CA subunit destination plug

The CA subunit destination plug is the input to the subunit. For DVB compliant systems the isochronous signal input to the subunit shall be compliant with the MPEG–2 specification [2]. The CA subunit destination plug can connect either directly to the serial bus (1394) input plug or to the source plug of another suitable subunit; for example the input to the CA subunit could be a DVB compliant tuner subunit.

5.1.2 CA subunit source plug

The CA subunit source plug is the output of the subunit. For DVB compliant systems the output signal from the subunit shall be compliant with the MPEG–2 specification. The CA subunit source plug can connect either directly to the serial bus output plug or to the destination plug of another suitable subunit.

5.1.3 Connections

When making connections between the CA subunit destination plug and either the serial bus input or another subunit the connection must be established manually using a CONNECT command. This connection must be made before issuing a CA command. If the CA subunit is operating in a stand-alone mode then the destination and source plugs of the subunit can be permanently connected to the input and output of the serial bus plugs.

If the CA subunit has an existing connection which has been locked and an additional connection is requested then a response of REJECTED shall be returned. If the connection is permanent then the conflicting command shall generate a response of NOT IMPLEMENTED.

It is necessary to use the CONNECT command to connect the CA subunit source plugs to either another subunit or the serial bus output plugs.

All current connections of CA subunits shall be reported by the CONNECT status or CONNECTIONS status commands. This includes all permanent connections. A controller can determine if a connection is permanent by examining the "perm" flag of the responses for the CONNECT status and CONNECTIONS status commands.

The connection of the CA subunit to other subunits is implementation specific. Whether it is logical to allow the connection of the CA subunit to certain other subunits should be considered at implementation time.

The normal scenario for the implementation of a CA subunit is to either have the subunit inside a receiver, which is a device defined as one that contains a tuner subunit, or as a stand-alone device.

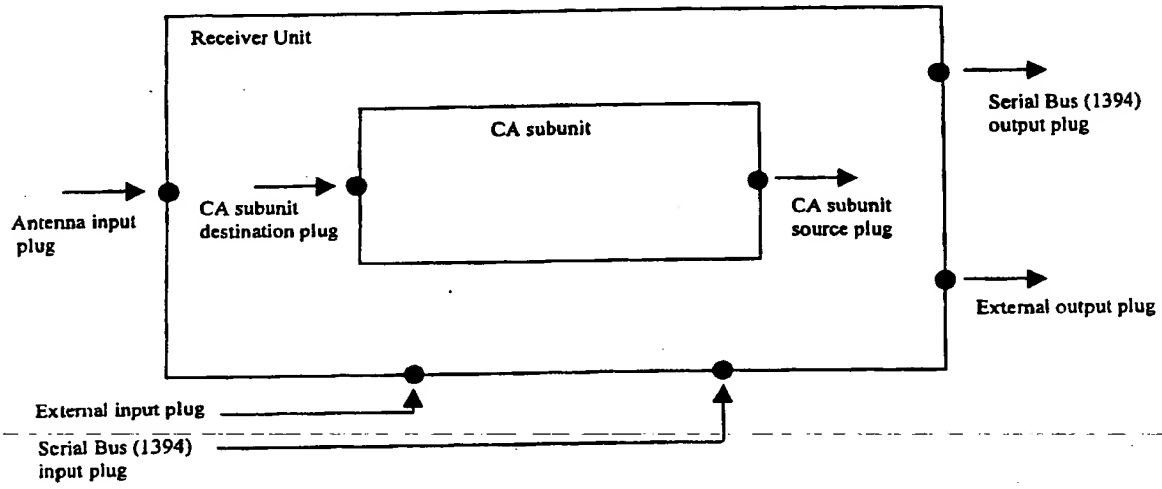

Figure 5-1 CA Subunit Logical Connections

The following table illustrates the various combinations of connections between a receiver unit and a CA subunit plugs and which ones are valid or not. All invalid connections shall generate a response of NOT IMPLEMENTED.

Non CA Subunit Plug	CA Subunit Plug	Connection Valid ?	Comments
External antenna input plug	CA destination plug	NO	X
External antenna input plug	CA source plug	NO	X
External input plug	CA destination plug	NO	X
External input plug	CA source plug	NO	X
External output plug	CA destination plug	NO	X
External output plug	CA source plug	NO	X
Serial bus input plug	CA destination plug	YES	This connection must be created using a CONNECT command, or it may be a permanent connection
Serial bus input plug	CA source plug	NO	X
Serial bus output plugs	CA destination plug	NO	X
Serial bus output plugs	CA source plug	YES	This connection must be created using a CONNECT command, or it may be a

			permanent connection
Subunit source plug	CA destination plug	YES	This connection must be created using a CONNECT command, or it may be a permanent connection
Subunit source plug	CA source plug	NO	X
Subunit destination plug	CA destination plug	NO	X
Subunit destination plug	CA source plug	YES	This connection must be created using a CONNECT command, or it may be a permanent connection

Table 5-1 Allowed Connections to a CA Subunit

5.1.4 Data Transmitted Over an Isochronous Channel

The controller should handle the transport stream intelligently. The IHDN is of limited bandwidth, so partial transport streams should be passed to the CA subunit. The controller must ensure that all tables are suitably modified to indicate that the stream is a partial transport stream. Some of the data carried to enable conditional access is carried in private data fields. Hence the controller must not send only the relevant audio and video channels.

5.2 *CA Subunit Identifier Descriptor, Status Descriptor, Objects and Object Lists*

The CA subunit will use the defined general subunit identifier descriptor. The general descriptor is fully explained in the AV/C Digital Interface Command Set General Specification, Version 3.0FC1, March 29, 1998.

CA Subunit Identifier	
Address offset	**Contents**
00_{16}	descriptor_length
01_{16}	
02_{16}	generation_ID
03_{16}	size_of_list_ID
04_{16}	size_of_object_ID
05_{16}	size_of_object_position
:	number_of_root_object_lists(n)
:	
:	root_object_list_id_0
:	
:	:
:	root_object_list_id_n-1
:	
:	
xx xx_{16}	subunit_dependent_length
:	
:	subunit_dependent_information
:	
xx xx_{16} + Subunit_ dependent_length + 1	
yy yy_{16}	manufacturer_dependent_length
:	manufacturer_dependent_information
yy yy_{16} + manufacturer_	

CA Subunit Identifier	
dependent_length	

Table 5-1 General Subunit Identifier Descriptor

The *descriptor_length* filed contains the number of bytes used for this descriptor structure.

The *generation_ID* field specifies which AV/C descriptor format is used by this subunit for all data structures it maintains, and the command sets which affect them.

The *size_of_list_ID* field indicates the number of bytes used to indicate a list ID for this subunit. All objects maintained within the scope of the subunit which have an ID shall use this number of bytes for their ID.

The *size_of_object_ID* field indicates the number of bytes used to indicate an object ID for this subunit.

The *size_of_object_position* field indicates the number of bytes used when referring to an object by its position in a list.

The *number_of_root_object_lists* field contains the number of root object lists associated with this subunit.

The *root_object_list_id_x* fields are the ID values for each of the root object lists associated with this subunit. The *number_of_root_object_lists* field indicates how many of these ID values are present.

The *subunit_dependant_length* and *subunit_dependent_information* contain information whose format and contents are described in section 5.2.1.

The *manufacturer_dependent_length* and *manufacturer_dependent_information* are used for vendor specific data. If there is no manufacturer dependant information in the descriptor, then the length field shall be set to zero and the *manufacturer_dependent_information* field shall not exist.

5.2.1 CA Subunit Identifier

A CA subunit shall have the same basic SUBUNIT IDENTIFIER DESCRIPTOR structure as presented in the section titled The General Subunit Identifier Descriptor in the AV/C Tuner subunit document [7].

The CA subunit shall have the following *subunit_dependent_information* within the subunit identifier descriptor.

CA Subunit Identifier	
Address offset	**Contents**
00_{16}	specification_length
01_{16}	
02_{16}	system_id
03_{16}	implementation_profile_id
04_{16}	number_of_CA_system_ids(m)
05_{16}	CA_system_id_length[0]
	CA_system_id[0]
:	
:	
:	CA_system_id_length[m-1]
:	CA_system_id[m-1]
:	

CA Subunit Identifier	
:	

Table 5-1 CA Subunit Identifier

The *specification_length* field indicates the size, in bytes of the entire descriptor structure.

The *system_id* field indicates a broadcast system that the CA subunit supports. The following broadcast systems are currently defined:

system_id	name
20_{16}	DVB(Digital Video Broadcast)

Table 5-2 System ID Codes

The *implementation_profile_id* field specifies the profile ID of the CA subunit for this *system_id*. A CA subunit may be implemented with a different profile for each of the systems that it supports. There shall be one profile for each supported system. The following profiles are currently defined:

implementation_profile_id	meaning
$F1_{16}$	conformant_full_implementation – a CA subunit with this profile i implements all of the commands and relevant data structures for the specified broadcast system, as defined in the AV/C CA subunit Working Specification version 1.0.
All others	reserved for future specification

Table 5-3 Implementation Profiles

The *number_of_CA_system_ids* field indicates the number of CA systems the CA subunit is compatible with.

The *CA_sytem_id_length* and *CA_system_id* fields can identify a particular CA system. For DVB systems the values for *CA_system_id* are defined in [6]. The label length field defines the length in bytes of the *CA_system_id* field.

5.2.2 CA Subunit Status Descriptor

The CA status descriptor structure is specific to CA subunits. It holds information about the CA subunit in general, and about what information is on each of its source plugs. The data held within this structure is dynamic and is kept up to date by the CA subunit. A controller may examine this structure in order to determine the operational status of the CA subunit and its source plugs.

The general format of the CA status descriptor is as follows:

CA Status Descriptor	
address	**contents**
$00\ 00_{16}$	descriptor_length
$00\ 01_{16}$	
$00\ 02_{16}$	general_CA_status
:	
:	

Table 5-1 CA Status Descriptor

The *descriptor_length* is the number of bytes for the entire CA subunit status descriptor structure.

The *general_CA_status* field contains information about the CA subunit. The *general_CA_status* field has this format:

general_CA_status format	
address offset	contents
00_{16}	general_CA_status_length
01_{16}	in_use
:	resource_location

Table 5-2 General CA Status Format

The *general_CA_status_length* is the number of bytes for the *general_CA_status* descriptor structure.

The *in_use* field describes whether the CA subunit is currently being used by a host, a one in this field signifies the subunit is being used and a zero signifies that the subunit is free to be used.

The *resource_location* field indicates whether the CA subunit relies on private internal resources such as a modem or requires external subunits. A value of "0" in this field indicates that the resources are private and a "1" indicates that external resources are required.

5.2.2.1 Descriptor Identifier for the CA subunit Status Descriptor

The CA subunit Status descriptor is specific to the CA subunit type, it has the following type value.

descriptor_type	meaning
TBD	CA Status Descriptor

Table 5-1 CA Subunit Status Descriptor Identifier

The descriptor_type_specific_reference field does not exist because there is only one CA status descriptor for a tuner subunit.

5.3 *CA Subunit Commands*

CA subunit commands are identified by a subunit_type of 06_{16} (***provisional***). This section defines how the commands are used.

		Support level (by ctype)			
Opcode	Value	C	S	N	Comments
CA_ENABLE	CC_{16}	M			Used to instruct the CA subunit to begin descrambling the service defined in the CA_PMT descriptor.

Table 5-1 DVB CI Subunit Commands

5.3.1 CA_ENABLE

The basic CA_ENABLE command is defined in the AV/C Tuner subunit specification. In the tuner subunit specification the CA_ENABLE command is defined as being system specific. The CA_ENABLE command is fully specified here for the DVB system.

For a DVB compliant system the CA_ENABLE command will take the following form.

	msb							lsb
opcode	CA_ENABLE (CC_{16})							
operand[0]	length(n)							
operand[1]	more_last							
operand[2]	frame_identifier							
:	system_id							
:	ca_pmt_list_management							
:	program_number							
:								
:	reserved	version_number						CNI
:	reserved	program_info_length						
:								
:	ca_pmt_cmd_id							
:	number_of_program_CA_descriptors[m]							
:	length_of_program_CA_descriptor[0]							
:	:							
:	program_CA_descriptor[0]							
:	:							
:	length_of_program_CA_descriptor[m-1]							
:	:							
:	program_CA_descriptor[m-1]							
:	:							
:	stream_type							
:	reserved	elementary_PID						
:								
:	reserved	ES_info_length						
:								
:	ca_pmt_cmd_id							
:	number_of_elementary_CA_descriptors[m]							
:	length_of_elementary_CA_descriptor[0]							
:	:							
:	elementary_CA_descriptor[0]							
:	:							
:	length_of_elementary_CA_descriptor[m-1]							
:	:							
:	elementary_CA_descriptor[m-1]							
:	:							

Figure 5-1 CA_ENABLE Control Command

The *length* field contains the byte length of the entire conditional access command block minus the length of the opcode and first three operands. So in the case where the data is split over several frames upon reassembly the first four bytes of the second to n-1 frames are discarded.

In some cases the data that the CA_ENABLE command carries may exceed the maximum length of an AV/C frame. In this case the *more_last* field is used to indicate whether all the data is contained in one frame or carried over to another frame. A value of "1" indicates that there is more data to follow and a value of "0" indicates that this is the last frame. If all the data will fit into one frame *more_last* takes the value of "0".

The *frame_identifier* field provides a means for controllers and targets to match fragmented frames. Whilst the *more_last* field is set to "1" frames with an identical *frame_identifier* value will be assembled together at the target. When a frame is received with *more_last* set to "0" it indicates that

this is the final frame with the current *frame_identifier* value and further frames received with the same *frame_identifier* are to be treated as new set of frames.

The system_id field denotes which broadcast system the following command relates to. The following stystem is defined so far.

system_id	name
20_{16}	DVB(Digital Video Broadcast)

Table 5-1 System ID Codes

The remaining entries in the CA_ENABLE descriptor are fully explained in [2] sections 2.4.4.8 and 2.4.4.9. However a brief description is included below.

The ca_pmt_list_management field indicates whether the user has selected a single program (made of one or several elementary_streams) or several programs. **The definition is made in the DVB Common Interface specification (A017, May 1996), section 8.4.3.4 and is repeated here.** The following values are defined.

ca_pmt_list_management	value
More	00
First	01
Last	02
Only	03
Add	04
Update	05
reserved	Other values

Table 5-2 Definition of ca_pmt_list_management

When set to 'first' it means that the CA PMT is the first one of a new list of more than one CA PMT object. All previously selected programmes are being replaced by the programmes of the new list. 'more' means that the CA PMT is neither the first one, nor the last one of the list. 'last' means that the CA PMT object is the last of the list. 'only' means that the list is made of a single CA PMT. 'add' means that this CA PMT has to be added to an existing list, that is, a new programme has been selected by the user, but all previously selected programmes remain selected. If 'add' is received for an already existing programme then its action is identical to 'update'. 'update' means that the CA PMT of a programme already in the list is sent again because the version_number or the current_next_indicator has changed. It is the responsibility of the application to check whether the change of the version_number results in a change of the CA operations or not. Since the list management commands only act at the programme level, any changes at the elementary stream level in an existing programme must be signalled by an 'update' command with the complete elementary stream list re-sent.

The *program_number* field specifies the program to which the program_map_PID is applicable.

The *version_number* field is the version number of the TS_program_map_section.

The *CNI* field is set to "1" to indicate that the TS_program_map_section sent is currently applicable. When set to "0" it indicates that the TS_program_map_section is not yet applicable and shall be the next TS_program_map_section to become valid.

The program_info_length specify the number of bytes of the descriptors immediately following the program info_length field.

The ca_pmt_cmd_id field is specified in the DVB Common Interface Specification [1], section 8.4.3.4 and is repeated here. The field indicates the required response from the application to a CA PMT object. It can take the following values.

ca_pmt_cmd_id	ca_pmt_cmd_id_value
Ok_descrambling	01
Ok_mmi	02
Query	03
Not_selected	04
RFU	Other values

Table 5-3 Definition of ca_pmt_cmd_id

When set to 'ok_descrambling' it means that the host does not expect answer to the CA PMT and the application can start descrambling the programme or start an MMI dialogue immediately. When set to 'ok_mmi' it means that the application can start a MMI dialogue but shall not start descrambling before reception of a new CA PMT object with ca_pmt_cmd_id set to 'ok_descrambling'. In this case the host shall guarantee that a MMI session can be opened by the CA application. When set to 'query' it means that the host expects to receive a CA PMT Reply. In this case, the application is not allowed to start descrambling or MMI dialogue before reception of a new CA PMT object with the ca_pmt_cmd_id set to 'ok_descrambling' or to 'ok_mmi'. When set to 'not_selected' it indicates to the CA application that the host no longer requires that CA application to attempt to descramble the service. The CA application shall close any MMI dialogue it has opened.

There then follows a series of CA_descriptors, the format of which is explained below. The CA_descriptor is specified in [2] section 2.6.16.

:	descriptor_tag
:	descriptor_length
:	CA_system_ID
:	
:	reserved | CA_PID
:	
:	private_data_byte [1]
:	:
:	private_data_byte [n-1]

Figure 5-2 CA_descriptor at Programme Level

The *descriptor_tag* field signifies that this is a CA_descriptor.

The *descriptor_length* field gives the length of the CA_descriptor.

The *CA_system_ID* field identifies the CA system in use.

The *CA_PID* field points to packets containing program related access control information, such as ECMs.

The remaining bytes of the CA_descriptor contain private information.

The *stream_type* field identifies the type of program element carried within the packets with the PID whose value is specified by the elementary _PID. The values are defined in table 2–29 of [2].

The *elementary_PID* field specifies the PID of the transport stream packets that carry the associated program element.

The *ES_info_length* field specifies the number of bytes of the descriptors of the associated program element immediately following the *ES_info_length*

The *ca_pmt_cmd_id* field is repeated here at the ES level.

There then follows the CA_descriptor for elementary stream level. The fields are the same as for the program level descriptor except for the *CA_PID* field, which now points to packets containing system wide, and/or access control management information, such as EMMs.

:	descriptor_tag
:	descriptor_length
:	CA_system_ID
:	
:	reserved | CA_PID
:	
:	private_data_byte [1]
:	:
:	private_data_byte [n-1]

Figure 5-3 CA_descriptor at Elementary Stream Level

In the case that the CA_ENABLE command is successful the response will be ACCEPTED. The response will have the following format.

	msb						lsb
opcode	CA_ENABLE (CC_{16})						
operand[0]	length(n)						
operand[1]	more_last						
:	frame_identifier						
:	program_number						
:							
:	reserved				version_number		CNI
:	CAEF	CA_enable (programme level) or reserved					
:	number_of_elementary_CA_enable_fields[m]						
:	elementart_CA_enable_field [1]						
:	:						
:	elementary_CA_enable_field [m-1]						

Figure 5-4 CA_ENABLE response

The fields, lengh, more_last and frame_identifier will have the same meaning as those of the same name in the CA_ENABLE control command.

The *program_number* field specifies the program to which the program_map_PID is applicable.

The *version_number* field is the version number of the TS_program_map_section.

The *CNI* field is set to "1" to indicate that the TS_program_map_section sent is currently applicable. When set to "0" it indicates that the TS_program_map_section is not yet applicable and shall be the next TS_program_map_section to become valid.

The response contains one possible *CA_enable* at programme level and for each elementary stream, one possible *CA_enable* at elementary stream level.

Note: When both are present only *CA_enable* at ES level applies for that elementary stream. When none is present, the controller does not interpret the CA_ENABLE response object.

The *CAEF* field signifies that the *CA_enable* field is valid, when set to "1" the following field is *CA_enable* and when set to "0" the following field is reserved.

CA_enable can take on the following values.

CA_enable	Value
Descrambling possible	01
Descrambling possible under conditions (purchase dialog)	02
Descrambling possible under conditions (technical dialog)	03

Table 5-4 Definition of CA_enable

The elementary_CA_enable_field has the following format.

:	reserved	elementary_PID
:		
:	CAEF	CA_enable (elementary stream level) or reserved

Figure 5-5 Format of elementary_CA_enable_field

The *elementary_PID* field signifies the elementary stream PID.

The *CAEF* and *CA_enable* fields have the same definition as above.

In the case where the CA_ENABLE command is unsuccessful the response frame will use the response code of REJECTED. The *CA_enable* field will take on the following values to reflect the nature of the error.

CA_enable	Value
Descrambling not possible (because no entitlement)	71
Descrambling not possible (for technical reasons)	73

Table 5-5 Definition of result field

The CA_ENABLE command can also be sent with a ctype of STATUS and NOTIFY as described in the tuner specification, in this case the response will be in the same format as the response from described above.

6. Smart Card Reader Commands

6.1 *Requirements*

The smart card subunit provides a set of asynchronous commands that are used to access and control a smart card. The smart card subunit relies on the asynchronous plug concept (under definition) to limit the number of controllers that can access the subunit.

The AV/C commands described below allow a controller to power on and off and reset the smart card. The status of the smart card can also be determined. The CONTROL commands for the smart card can only be accepted from the controller that has reserved the smart cart subunit. Commands from other controllers shall be REJECTED. STATUS commands from other controllers will be supported.

For sending a receiving data from the smart card the AV/C asynchronous file transfer protocol (under definition) shall be used. The exact nature of the transfer is to be defined. The data should be transmitted in a secure method to avoid exposing private data.

6.2 *Commands*

The first format of the FUNCTION command shall have a ctype value of CONTROL and is illustrated in figure XXX below:

	msb						lsb
opcode	FUNCTION						
operand[0]	smartcard_operation						

Figure 6-1 FUNCTION Control Command

smartcard_operation can take the values defined in table 6–1 below:

smart_card_operation	Value	Support level
power_on_card	TBD	M
power_off_card	TBD	M
reset_card	TBD	M
read_answ_to_reset	TBD	M
reserved	TBD	M

Table 6-1 Definition of smartcard_operation

power_on_card : switch on the card interface by connection and activation of the contacts by the interface device (Vcc then RAZ according to ISO 7816 specification). The response is ACCEPTED if a card is inserted and has performed a reset correctly and REJECTED if there is no card, or the card did not reset correctly.

power_off_card : powers off the card interface according to the ISO 7816 specification. The response is ACCEPTED if the operation is successful and REJECTED if the operation fails.

reset_card : resets the card interface when the card is powered on. The responses are the same as for *power_on_card.*

The second format of the SMARTCARD command, determining the status of the smart card, shall have a ctype STATUS and is illustrated in figure 6–2 below:

	msb						lsb
opcode	FUNCTION						
operand[0]	smartcard_operation						

Figure 6-2 SMARTCARD Status Command

smartcard_operation can take the values defined in table XXX below:

smartcard_operation	Value	Support level
card_status	TBD	M

Table 6-2 Definition of smartcard_operation

Used to find the current connection and card status.

The response to the SMARTCARD status command is illustrated in figure 6–3 below:

	msb						lsb
opcode	SMARTCARD						
operand[0]	smartcard_operation						

Figure 6-3 SMARTCARD status command response

smartcard_operation can take the values defined in table 6–3 below:

smartcard_operation	Value
card_inserted	TBD
card_removed	TBD
card_in_place_power_off	TBD
card_in_place_power_on	TBD
no_card	TBD
unresponsive_card	TBD
refused_card	TBD
reserved	TBD

Table 6-3 Definition of smartcard_operation response

card_inserted : sent as a response to a NOTIFY command when a card has been inserted.

card_removed : sent as a response to a NOTIFY command when a card is removed.

card_in_place_power_off : response when a card is inserted and recognised and the card is not powered.

card_in_place_power_on : response when a card is inserted and recognised and the card is powered.

no_card : response when no card is plugged in.

unresponsive_card : response when the plugged in card does not reply to a reset.

refused_card : response when a card responds to a reset, but not in the expected way.

7. Date & Time Resource

The NCAM may require date & time information. This information can be provided by a subunit implementing the date & time descriptor (under discussion) or the CA subunit may implement the date & time function privately in an implementation specific manor, for example by implementing a real time clock.

8. Modem Subunit

8.1 *Requirements*

The modem subunit should provide bi-directional low speed communication over a suitable medium such as a telephone line or cable network return channel. The subunit can be used to support CA applications and interactive services.

The subunit should implement the AV/C asynchronous file transfer protocol (under definition) for transferring data across the 1394 network. The subunit shall implement the asynchronous plug concept to ensure secure data paths across the network. The number of plugs the modem can support shall equal the number of simultaneous connections it can support. Full duplex communication should be possible. The subunit should be defined in a generic way to allow different underlying communication methods to be used.

8.2 *Connections*

The modem subunit relies on the asynchronous plug concept to ensure that the integrity of the connections to the subunit are maintained. The exact nature of the connection functionality is under definition and this section will be expanded at a later date to explain the connection strategy. The asynchronous plugs will allow a dedicated asynchronous connection to be made to the modem subunit. Only connected controllers will be able to issue CONTROL commands to the modem subunit, the subunit will accept STATUS commands from non connected controllers.

8.3 *Modem subunit Identifier Descriptor, Status Descriptor, Objects and Object List*

The Modem subunit will use the defined general subunit identifier descriptor.

General Subunit Identifier	
Address offset	Contents
00_{16}	descriptor_length
01_{16}	
02_{16}	size_of_object_ID
03_{16}	number_of_root_object_lists(n)
04_{16}	root_object_list_id_0
05_{16}	
:	:
	root_object_list_id_n-1
:	
:	subunit_dependant_length
xx xx_{16}	
:	subunit_dependant_information
:	
:	
xx xx_{16} + Subunit_ dependant_length + 1	manufacturer_dependent_length
yy yy_{16}	
:	manufacturer_dependent_information
:	

General Subunit Identifier	
yy yy_{16} + manufacturer_dependant_length	

Figure 8-1 General Subunit Identifier Structure

The *descriptor_length* filed contains the number of bytes used for this descriptor structure.

The *size_of_object_ID* field indicates the number of bytes used to indicate an object ID for this subunit.

The *number_of_root_object_lists* field contains the number of root object lists associated with this subunit.

The *root_object_list_id_x* fields are the ID values for each of the root object lists associated with this subunit. The *number_of_root_object_lists* field indicates how many of these ID values are present.

The *subunit_dependant_length* and *subunit_dependant_information* contain information whose format and contents are described in section 5.1.

The *manufacturer_dependent_length* and *manufacturer_dependent_information* are used for vendor specific data. If there is no manufacturer dependant information in the descriptor, then the length field shall be set to zero and the *manufacturer_dependent_information* field shall not exist.

8.3.1 Modem Subunit Identifier

The Modem subunit shall have the same basic SUBUNIT IDENTIFIER DESCRIPTOR structure as presented in the section titled The General Subunit Identifier Descriptor in the AV/C Tuner subunit document [5].

The Modem subunit shall have the following *subunit_dependant_information* within the subunit identifier descriptor.

Modem Subunit Dependent Information	
Address offset	Contents
00_{16}	number_of_systems [n]
01_{16}	system[0]_specification
:	:
:	system[n-1]_specification
:	
:	

Figure 8-1 Modem Subunit Dependent Information

The *number_of_systems* field specifies how many transport technologies are supported by this modem subunit.

The *system[x]_specification* field describes a modem technology, it has the following format.

Modem Subunit Identifier	
Address offset	Contents
00_{16}	specification_length
01_{16}	modem_type
02_{16}	implementation_profile_id

Figure 8-2 Modem Subunit Identifier

The *specification_length* field indicates the size, in bytes of the entire descriptor structure.
The *modem_type* field indicates the underlying technology that the modem subunit supports. The following technologies are currently defined:

modem_type	Technology
01_{16}	PSTN Modem
02_{16}	Cable Return Channel Modem

Table 8-1 Definition of modem_type

The *implementation_profile_id* field specifies the profile ID of the modem subunit for this *modem_type*. A modem subunit may be implemented with a different profile for each of the modem types that it supports. There shall be one profile for each supported modem type. The following profiles are currently defined:

implementation_profile_id	meaning
$F1_{16}$	conformant_full_implementation – a modem subunit with this profile implements all of the commands and relevant data structures for the specified modem technologies, as defined in this current draft.
All others	reserved for future specification

Table 8-2 Definition of implementation_profile_id

8.3.2 Modem Subunit Status Descriptor

The modem status descriptor structure is specific to modem subunits. It holds information about the modem subunit in general. The data held within this structure is dynamic and is kept up to date by the modem subunit. A controller may examine this structure to determine the operational status of the modem subunit.

The general format of the CA status descriptor is as follows:

Modem Status Descriptor	
address	**contents**
$00\ 00_{16}$	descriptor_length
$00\ 01_{16}$	
$00\ 02_{16}$	general_modem_status
:	
:	

Figure 8-1 Modem Status Descriptor

The *descriptor_length* is the number of bytes for the entire modem subunit status descriptor structure.

The *general_modem_status* field contains information about the modem. *The general_modem_status* field has this format:

general_modem_status format	
address offset	**contents**
00_{16}	general_modem_status_length
01_{16}	in_use

Figure 8-2 General Modem Status Format

The general_modem_status_length field gives the length of the general_modem_status descriptor.

The in_use field is a 1-bit flag that takes the value of "1" when the modem subunit is in use and "0" when the modem subunit is available.

8.3.2.1 Descriptor Identifier for the CA subunit Status Descriptor

The modem subunit Status descriptor is specific to the modem subunit type, it has the following type value.

descriptor_type	meaning
TBD	modem Status Descriptor

Table 8-1 modem Subunit Status Descriptor Identifier

The descriptor_type_specific_reference field does not exist because there is only one modem status descriptor for a tuner subunit.

8.4 *Modem Subunit Commands*

The subunit type for the modem subunit is to be defined.

8.4.1 connect_PSTN_modem_channel

The *connect_PSTN_modem_channel* command is issued to the modem subunit to instruct a PSTN modem to establish a connection. The connect_PSTN_modem_channel has a ctype of CONTROL.

address	msb							lsb
opcode	connect_PSTN_modem_channel							
operand0]	retry_count							
operand[1]	timeout							
:	reserved			FA	connection_type			
:	reserved	CPL			IACL			OCL
:	reserved	national_area_code_length			core_number_length			
:	: country_prefix_char :							
:	: international_area_code_char :							
:	: operator_code_char :							
:	: national_area_code_char :							
:	: core_number_char :							

Figure 8-1 connect_PSTN_modem_channel Control Command

The *retry_count* field signifies the number of retries to be attempted.

The *timeout* field indicates the maximum number of seconds a connection attempt should remain valid. If there is no positive indication of the state of the connection within the time *timeout* the connection will be aborted. A *timeout* value of zero means wait indefinitely.

The remaining fields are taken from the DVB SI Telephone descriptor which are defined it [6] section 6.2.30. A summary of the fields is given below.

All *reserved* fields are to have a value of 00_{16}.

The *FA* field is a 1-bit flag which when set to "1" signifies that the number described can be called from outside of the country. When set to "0" the number can only be called from inside the country.

The *connection_type* field indicates connection types.

The *CPL* field specifies the number of 8-bit alphanumeric characters in the country prefix.

The *IACL* field specifies the number of 8-bit alphanumeric characters in the international area code.

The *OCL* field specifies the number of 8-bit alphanumeric in the operator code.

The *national_area_code_length* field specifies the number of 8-bit alphanumeric characters in the national area code.

The *core_number_length* field specifies the number of 8-bit alphanumeric characters in the core number.

The *country_prefix_char* field, which shall be encoded in accordance with ISO 9958-1 [7], gives one alphanumeric character of the country prefix.

The *internationnal_area_code_char* field, which shall be encoded in accordance with ISO 9958-1 [11], gives one alphanumeric character of the international area code.

The *operator_code_char* field, which shall be encoded in accordance with ISO 9958-1 [11], gives one alphanumeric character of the operator code.

The *national_area_code_char* field, which shall be encoded in accordance with ISO 9958-1 [11], gives one alphanumeric character of the national area code.

The *core_number_char* field, which shall be encoded in accordance with ISO 9958-1 [11], gives one alphanumeric character of the core number.

The connect_PSTN_modem_channel has the following responses.

If the connection is successful the response is ACCEPTED, and has the following format.

	msb							lsb
opcode	connect_PSTN_modem_channel							
operand[0]	connection_result							

Figure 8-2 connect_PSTN_modem_channel ACCEPTED Response

connection_result	Description
connected	connection okay

Table 8-1 Definition of connection_result

If the connections fails a response of REJECTED will be returned. The response will have the following format.

	msb							lsb
opcode	connect_PSTN_modem_channel							
operand[0]	connection_result							
operand[1]	reason							

Figure 8-3 connect_PSTN_modem_channel REJECTED Response

connection_result	Description
retry_fail	exceeded retry count
timeout	exceeded timeout

Table 8-2 Definition of connection_result

reason	Description
unexplained	an unexplained fault occurred
engaged	target is engaged
unobtainable	number is unobtainable

Table 8-3 Definition of reason

8.4.2 connect_cable_modem_channel

The connect_cable_modem_channel command is issued to the modem subunit to instruct a cable return channel modem to establish a connection. The command has a ctype of CONTROL.

	msb							lsb
opcode	connect_cable_modem_channel							
operand[0]	retry_count							
operand[1]	timeout							
operand[2]	channel_id							

Figure 8-1 connect_cable_modem_channel Control Command

The fields retry_count and timeout fields are the same as for the connect_PSTN_modem_channel command.

The channel_id field defines the return channel ID number.

The connect_cable_modem_command has the following responses.

If the connection is made then the response is ACCEPTED and has the following format.

	msb							lsb
opcode	connect_cable_modem_channel							
operand[0]	connection_result							

Figure 8-2 connect_cable_modem_channel ACCEPTED Response

connection_result	Description
connected	connection okay

Table 8-1 Definition of connection_result

If the connections fails a response of REJECTED will be returned. The response will have the following format.

	msb						lsb
opcode	connect_cable_modem_channel						
operand[0]	connection_result						
operand[1]	reason						

Figure 8-3 connect_cable_modem_channel REJECTED Result

connection_result	Description
retry_fail	exceeded retry count
timeout	exceeded timeout

Table 8-2 Definition of connection_result

reason	Description
unexplained	an unexplained fault occurred

Table 8-3 Definition of reason

Note: If a connect_PSTN_modem_channel or connect_cable_modem_channel command is sent to a modem that does not implement the technology described in the command a response of NOT IMPLEMENTED will be returned.

8.4.3 disconnect_modem_channel

	msb						lsb
opcode	disconnect_modem_channel						
operand[0]	FF_{16}						

Figure 8-1 disconnect_modem_channel Control Command

The disconnect_modem_channel command is used to terminate the current connection. If the command is successful a response of ACCEPTED is returned, otherwise REJECTED is returned.

8.4.4 modem_status

It is also possible to request the status of the modem subunit. The information that can be returned is defined in the modem status descriptor. The modem_status command can have a ctype of STATUS and NOTIFY.

	msb						lsb
opcode	modem_status						
operand[0]	FF_{16}						

Figure 8-1 modem_status Status Command

8.4.5 modem_data_transfer

The commands for transferring data to and from the modem are to be defined. The exact nature of these commands depends on the outcome of the AV/C asynchronous file transfer discussion.

9. Operational Guidelines

The following provides an explanation as to how the NCAM will be implemented and the procedure that shall be followed to make use of the NCAM.

The NCAM is a logical collection of subunits that provide the required functionality to implement a networked conditional access system. The CA subunit is the core of the system and relies on other subunits to provide a source and sink for the material that requires descrambling and communication with both the user and outside world. As such the CA subunit must be aware of the tuner subunit, panel subunit and modem subunit.

When connected to the bus and powered on the CA subunit will search the network and build a list of resources. The CA subunit will maintain this list whilst it is powered on. The CA subunit may accept status commands from any controller but control commands and writing to the CA subunit descriptors may only be executed once a control has made a connection to the CA subunit via the asynchronous plug control protocol (to be defined). This is to ensure that a controller has the exclusive use of the CA subunit during the length of the transaction.

The procedure for decoding a scrambled transport stream is as follows. The following assumes that the tuner subunit will be the source of the scrambled stream, either an off air signal via a suitable front end or directly from the demux via an alternative source such as a DVCR. The user will a make a channel selection and the tuner subunit will detect that the strcam is scrambled.

The controller can make an intelligent prediction as to which CA subunit to use based upon the *system_id* field from the transport stream and *system_id*. The controller then establishes an asynchronous connection with the CA subunit and writes the CA_PMT descriptor to the CA subunit. The controller establishes an isochronous channel to transmit the scrambles service to the CA subunit and a second channel from the CA subunit to the desired sink. When the controller has finished writing the descriptor it sends a CA_ENABLE command to instruct the CA subunit to commence descrambling.

When the CA subunit receives the CA_ENABLE command it determines whether or not it is capable of descrambling the selected service. This may involve setting up a dialog with the user to determine whether they are prepared to pay for the service and request them to insert their bank card or pin number. Some communication with the outside world via the modem may be required. The CA subunit updates the CA_PMT_REPLY descriptor as required and sends a response to the CA_ENABLE command.

The transmission of full transport streams across the network is wasteful of bandwidth and should be avoided. The controller should intelligently remove those audio and video channels that are not required and modifying the required tables to indicate the channels form a partial transport stream.

The NCAM does not follow the same concept of the DVB CI whereby isochronous data is continually passed between all connected modules. It is suggested that when a controller wishes to make a connection to a CA subunit to initiate a conditional access transaction and there are multiple CA subunits available the controller should first inspect the system_id field to identify a suitable CA subunit. If the controller wishes to try all available AV suunits it should broadcast the desired program to all CA subunits and then when a suitable CA subunit is located the program can be sent just to the relevant subunit.

10. Outstanding Points

The Panel subunit, asynchronous plug control mechanism, asynchronous file transfer and a date & time resource are all under discussion at present. These are key resources that are required for the efficient implementation of a NCAM. The exact utilisation of these resources in the NCAM will be included in this proposal when the exact definition of the resources is complete.

At present only the DVB Conditional Access system is provided for, at a later date other conditional access systems can be included. The modem types defined are the same as those supported by the DVB Common Interface, at a later other modem technologies may be defined.

The mechanism by which the user chooses the source and destination of the program needs to be defined. Although this is a general IHDN issue not confined to CA applications, indeed it may be implementation specific.

The use of the panel subunit for the MMI must be carefully implemented. The system must direct the MMI data to the display device relevant to the user. For example if the user selects the tuner subunit for the data source that is located in different room to their current position the MMI information must be routed back to the users location and not the tuner location.

PCT/GB99/01392

JAKEMP + CO

9/5/99

BLANK PAGE